OCR
A Level
Computer Science
Component 01

Data types

Unit 6

PG ONLINE

6

Teacher’s Guide

Introduction

This teacher’s guide contains a detailed lesson plan to accompany the set of PowerPoint slides and worksheets for each topic.

The lesson plans are designed to form a basis for ideas for the teacher and should be adapted to suit the teaching style and preferences of the individual teacher, and the resources and nature of the individual school or Computing / ICT department.

The material supplied for this unit includes:

- 5 PowerPoint presentations, each designed to cover one topic, which may take more than one lesson
- 5 worksheets
- 5 homework sheets
- An end-of-unit test for assessment purposes

Summary

This unit covers the representation of data in *Section 1.4.1* of the 2015 H446 specification. Five topics in this unit cover data representation of numbers and text, binary arithmetic using both fixed point and normalised floating point numbers, bitwise manipulation and masks. Each of the five topics may be spread over more than one lesson, especially if time is spent in the lessons going over homework tasks.

Learning Outcomes for the unit

At the end of this Unit all students should be able to:

- list primitive data types
- represent positive integers in binary
- use sign and magnitude and two's complement to represent negative numbers in binary
- add two unsigned binary numbers
- represent positive numbers in hexadecimal
- convert between denary, binary and hexadecimal number systems
- define bits and bytes, and use names, symbols and prefixes appropriately
- know how to use the ASCII table to represent text as binary
- explain why Unicode was introduced, and its advantages
- use arithmetic operations and Boolean operations AND, OR and XOR
- show the effect of a logical shift left and shift right of a number of bits

Most students will be able to:

- use fixed point binary to represent numbers with a fractional part
- convert a positive floating point number to denary and vice versa
- normalise a positive floating point number
- use arithmetic, logical and circular shifts
- differentiate between the character code for a digit and its pure binary representation

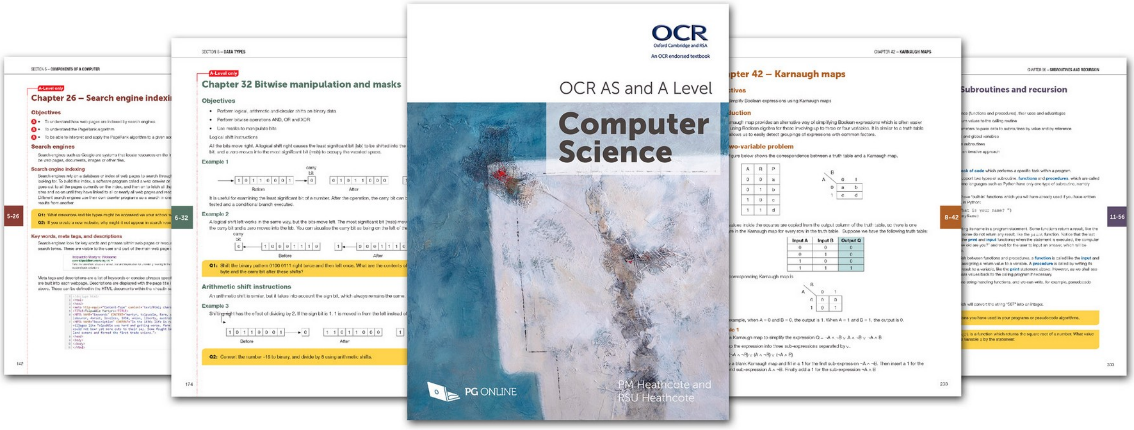
Some students will be able to:

- normalise negative floating point numbers
- add and subtract floating point numbers
- use shifts, bitwise manipulation and masks to solve problems

Previous Learning

Some exposure to data representation during Key Stages 3 and 4 will certainly help students to understand the more advanced concepts in this unit, but no previous learning is necessary.

Suggested Resources



The textbook *OCR A Level Computer Science* by PM Heathcote and RSU Heathcote provides comprehensive coverage of all the theoretical topics in the OCR specification (H446) and complements the unit series.

Each of the twelve sections in the book corresponds to one of the teaching units in this series and will be extremely useful as a course textbook and also as a revision guide. Sample questions, many taken from past exam papers, are included at the end of each chapter and can be set as additional homework.

The book is published by PG Online in a printed and electronic edition. Please refer to www.pgonline.co.uk for ordering and pricing details.

Vocabulary

Vocabulary associated with this Unit, such as:

Primitive data types, integer, real/floating point, character, string, Boolean, denary, binary, signed and unsigned, sign and magnitude, fixed point, floating point, two's complement, kibi, mebi, gibi, hexadecimal, ASCII, Unicode, character set, bitwise manipulation, mask, arithmetic shift, logical shift and circular shift

Assessment

Students will sit an end-of-unit test.

A few points to note:

These are not live assessment questions. They have all been created from scratch for this scheme of work. We cannot guarantee the areas covered in the test will cover all areas that could come up in any given exam paper. That being said, when producing the test the following have been carefully taken into account:

- the range of questions is designed to elicit the understanding of students from E-A* grade.
- appropriate command words and language is used across the range of questions (list, describe, state, discuss, explain…)

Real exam papers go through a serious quality assurance process; feel free to use and adapt these questions as you see fit.

Topic plans

Topic 1	Primitive data types, binary and hexadecimal
Learning Objectives: • List and define primitive data types • Represent positive integers in binary and hexadecimal • Convert between binary, hexadecimal and denary number systems	
Content	**Resources**
Starter Show the first slide, which defines a primitive data type. Students should be able to come up with Boolean, character and string. Without knowing what data type is held in a particular word in a computer's memory, it is not possible to say what a particular bit pattern represents. **Main** **Binary number system** Ask students to recap why counting is performed in groups of ten. Explain that this **denary** number system is referred to as **base 10** due to the fact it uses ten different symbols to count with. Extend this concept to a number system that uses only the symbols 0 and 1 to count with. Ask students to suggest what base this number system has and what it is commonly referred to as, (**base 2** or **binary**.) Demonstrate how to count in binary using the same principles when counting in tens by adding additional values to the left of previous numbers, (e.g. 1, 10, 11, 100.) Show that the position of the binary digits represents different values compared to denary. Work through the example of 183_{10} where each place value is a power of the base, (1 lot of 10^2, 8 lots of 10^1 and 3 lots of 10^0.) Compare this to the same process applied to 10110111_2 where again each place value is a power of the base, (1 lot of 2^7, 0 lots of 2^6, etc.) Demonstrate how to convert from denary to binary using a table of place values. Ask students to complete **Task 1** of **Worksheet 1**. Go over the answers when students have completed	PowerPoint Guide: Data types Topic 1 Data types, binary and hexadecimal.pptx

the task. **Hexadecimal** Introduce the concept of counting with **base 16**. Ask students to identify what base 10 actually means and how it is a problem for counting in 16s. Outline how the **hexadecimal system** works using letters for values past 9. Demonstrate how to count in hexadecimal using the same principles as counting in tens, (eg. E, F, 10, 11.) **Converting between hex and denary** Show that the position of hexadecimal digits represents different powers of 16. Work through the example $3F5_{16}$, (3 lots of 16^2,15 lots of 16^1, 5 lots of 16^0.) Demonstrate how to convert from denary to hex using a table of place values and that it uses the same process as binary and denary. $A3_{16} = 16 \times 10 + 3 = 163_{10}$ $27_{10} = 16 + 11 = 1B_{16}$ Ask students to identify the values of the first 4 powers of 2. Why is the fourth power significant? Note that 2^4 is 16 and that we have just learnt about a number system with a base of 16. This is convenient when converting between hex and binary as four bits of a binary value can be directly converted into an equivalent hexadecimal value. This can be repeated for all the bits of a binary value and the hex values are joined to give an overall value. Show how groups of 4 bits of a larger binary value are equivalent to one hex value and use the example to show how this can be used when converting between bases. Ask students to complete **Task 2** of **Worksheet 1**. **Plenary** Go over the answers to the worksheet with the class. Ask why hex is used. The students may well come up with some good answers. Show them the penultimate slide. Finally recap the lesson using the last slide. Give out the **Homework 1** sheet – Converting between hex and binary.	Data types Worksheet 1 Data types, binary and hexadecimal Data types Worksheet 1 Answers Data types Homework 1 Data types, binary and hexadecimal Data types Homework 1 Answers

Topic 2	Number systems, ASCII and Unicode

Learning Objectives:

- Define a bit as a 1 or a 0, and a byte as a group of eight bits
- Know that 2^n different values can be represented with *n* bits
- Use names, symbols and corresponding powers of 2 for binary prefixes e.g. Ki, Mi
- Differentiate between the character code of a denary digit and its pure binary representation
- Describe how character sets (ASCII and Unicode) are used to represent text

Content	Resources
Starter Show the students a simple electrical circuit with a switch and a light bulb. Ask them to explain how it works using the keywords provided with the circuit diagram. Highlight that when the switch is closed the light bulb turns on because the battery voltage is applied across it as the current flows through the circuit. Introduce the concept that, as a computer is an electrical device with millions of these switches connected by wires, data is stored and processed as a series of ON and OFF voltage signals. This is why the on/off symbol is a 0 and a 1. **Main** Link this back to the concept of binary covered in Topic 1. Recap that the denary value 21_{10} can be represented by the binary number 10101_2. Explain how this could be translated to electrical signals where a 1 value is represented by an ON voltage and 0 as OFF. Define each value in a binary number as a **bit** and show how the combination of bits could be represented as a collection of ON and OFF signals. Show that for *n* number of bits there are 2^n different combinations. Ask students to identify the combinations possible for 3 bits. These are: 000, 001, 010, 011, 100, 101, 110, 111. Note that this is also 2^3, or 8 combinations. Emphasise the simplicity of this approach. ON and OFF signals can be changed quickly and values easily determined. Computers can operate at high speeds as binary systems need only recognise *some voltage* and *no voltage*. **Bytes** Define a byte as 8 bits. Refer back to the previous	PowerPoint guide: Data types Topic 2 ASCII and Unicode.pptx

example of 10101_2. If this was represented as a byte of data it would need to be padded with leading zeros, giving 00010101_2. Large numbers of bytes can be referred to by using **prefixes**. Explain how prefixes work by describing how 1000 grams can be referred to as 1 kilogram where the kilo prefix represents 1000. Note that in the same way 1000 bytes can be referred to as a **kilobyte** or **kB**, (note that *B* refers to a byte and *b* refers to a bit.) Outline other common prefixes: **mega** (M) is 10^6; **giga** (G) is 10^9; **tera** (T) is 10^{12}. **Binary prefixes** Historically, computer scientists used these prefixes to refer to large numbers of bytes, so that a kilobyte meant 1024 bytes or "approximately 1000 bytes" The problem with this nomenclature was that the term **kilo** represented two different values in binary and denary number systems. In order to eliminate this confusion, in 1998 the International Electrotechnical Commission, (IEC,) established different prefixes to represent multiples of base 2: **kibi** (Ki) 2^{10}; **mebi** (Mi) 2^{20}; **gibi** (Gi) 2^{30}; **tebi** (Ti) 2^{40}. The difference between each prefix is 1024, (so 1 GiB is 1024 MiB, etc.) Ask students to complete **Worksheet 2**, **Task 1**. Go over the answers when students have completed the task. **Representing text** Link back to the need for computers to use ON and OFF signals. Introduce the concept of **character sets** whereby letter, numbers and symbols can be input and processed by a computer as a series of binary codes. Hand out the **ASCII Codes sheet**. Discuss the ASCII standard of using 7 bits to represent different characters. There are close to 100 different characters that can be typed using a standard keyboard – 128 characters can be represented using 7-bit ASCII. Ask students what limitations there are of using a 7-bit character set. 32 represents a space. Pressing ALT + 65 will produce the letter A (Use the numeric keypad.) **Character form of decimal digits** Note how the code 0111001 represents the character '9' but the binary byte representing this number would	Data types Worksheet 2 ASCII and Unicode Data types Worksheet 2 Answers ASCII Codes Character form of a decimal digit.py

be 00001001_2.
 	
This means you cannot treat digits as characters, and vice versa. In Python, all input from the user is assumed to be a string of one or more characters so if it is to be interpreted as an integer or real number, it must be converted using the function **int** or **float**.
 	
You can demonstrate using the Python program **Character form of a decimal digit.py** how a computer will react to the additional or concatenation of numeric digits depending on their form. The **ord()** function returns the decimal ASCII value. The program will output:
 	
`9 + 3 = 12`
`"9" + "3" = 93`
`ord("9") + ord("3") = 108`
 	
Unicode
 	
Explain how the number of characters are limited and other alphabets from other languages cannot be represented, (Greek or Russian letters, Chinese characters or letters with accents for example.) Although ASCII was extended to 8 bits where additional characters could be used beyond the original codes, different character sets for this extended range could be adopted but there was not one consistent standard. Many different countries used this extension for symbols in their own language called *code pages*. Problems occurred if a sender of a message used a different code page to the receiver.
 	
Introduce the **Unicode** scheme as a standardised alternative to ASCII. Note that Unicode can apply a variable length encoding from 1 byte up to 4 bytes. The benefit of Unicode is that the first 128 characters are the same as ASCII ensuring cross-compatibility. With 4 bytes, there are enough binary permutations that Unicode can accommodate all characters from all languages and other symbols or even emoticons such as ☺.	

The disadvantage of using 4 bytes per character is that text occupies more memory or storage space, and will take longer to transmit.	ASCII Conversion.xlsx
Ask students to complete **Task 2** of the worksheet to practise converting between ASCII and characters. You can use **ASCII Conversion.xlsx** to demonstrate.	Data types Homework 2 ASCII and Unicode Data types Homework 2 Answers
Discuss answers when students have completed the task.	

Plenary Go over Worksheet answers, summarise the topic and hand out **Homework 2**.	

Topic 3	Binary arithmetic

Learning Objectives:

- Use sign and magnitude to represent negative numbers in binary
- Use two's complement to represent negative numbers in binary
- Add and subtract binary integers
- Represent fractions in fixed point binary

Content	Resources
Starter Ask students to identify what computers can do with numerical values. Ensure that they note the basic **arithmetic**, (addition, subtraction, division, multiplication,) and **comparison** operations, (equal to, greater than, less than etc.) **Main** **Binary addition** Introduce binary addition. Explain that the method is basically the same as for simple denary addition: $19_{10} + 17_{10} = 36_{10}$, where the carry is used in the addition for the next place value. Extend this process to binary addition. Outline the rules of binary addition and demonstrate with the example $10101_2 + 111_2$ showing how each carry is used in addition into the next place value. Ensure that students understand how to add multiple bits by performing a series of additions with the result of each added to the next bit. You might point out that 1+1+1 is 3_{10} or 11_2, 1+1 is 2_{10} or 10_2 which will help students remember what to do in these 'carry' instances. Recap that binary in a computer system uses a fixed number of bits. Ask why the addition $11110000_2 + 10011111_2$ cannot be calculated using 8 bits. Note that for fixed number of bits the result of the addition will **overflow** at the end. The resulting answer will not be accurate unless a second byte is used as part of the calculation. Demonstrate this with the slide example. Complete **Task 1** on **Worksheet 3**. **Negative numbers: Sign and magnitude** Using this system, the most significant bit (msb) is used as a sign bit with 0 representing plus (+) and 1 representing minus (-). The problem with this system is that arithmetic does not work in the normal way, and the sign bit has to be handled separately. For this	PowerPoint guide: Data types Topic 3 Binary arithmetic.pptx Data types Worksheet 3 Binary arithmetic Data types Worksheet 3 Binary arithmetic Answers

reason, the alternative two's complement method is used.

Two's complement

Introduce the use of complementary values to represent negative values. Signed binary is used for negative values, or those with a sign, either + or -. In the **two's complement** system, the most significant bit (msb) is set to 1 to indicate a negative number, and takes a value of -128. This reduces the number of bits used for the actual value to seven, meaning that any number from -128_{10} to $+127_{10}$ can be represented. Show some examples of positive and negative values to illustrate this.

For example, 10000001 = -128 + 1 = -127

11000000 = -128 +64 = -64

Note that binary values can be converted between two's complement and standard binary by inverting each bit and adding one. For example 51_{10} which is 00110011_2 inverted becomes 11001100_2. With one added the two's complement value becomes 11001101_2 which is -51_{10}.

Highlight the formula for the range with n bits and ask students what the range is with 16 bits.

Demonstrate the use of two's complement when adding values using the same process as before. Show how 65_{10} + -43_{10} gives 22_{10}. Also prove that 65_{10} + 43_{10} still gives the correct answer of 108_{10}. Note that overflow errors can still distort the output if the calculated value cannot be held in the range.

Complete **Task 2** on **Worksheet 3**.

Binary fractions using fixed point

Introduce the concept of **fixed-point binary** numbers where a decimal place is assumed to be in a set position within the ranges of bits, (for example, in a byte of 8 bits, the decimal place could be between the fourth and fifth bits). Note how values to the right of the decimal point represent fractions. Each fraction is a negative power of 2, (ie: 2^{-1}, 2^{-2}, and so on). Show with the example of 6.625_{10} how the fixed-point binary byte can be determined if the number weights are known. Discuss how this process can be reversed to convert back to denary in the same way we convert standard binary values.

Ask students what effect adding a point has on the range and accuracy of representation of decimal values. Each value after the point will half the range of values. The greater the number of values after the point, the more accurately a fraction can be

represented, but many fractions such as fifths, cannot be represented at all. For this, you would require floating point binary which is not covered in the AS specification. Complete **Task 3** on **Worksheet 3**. Recap that the fundamental building block of processor operation is addition Note that two's complement enables subtraction to be performed by making values negative and fixed point binary allows fractions to be represented. Give out **Homework 2 Binary arithmetic**.	 Data types Homework 3 Binary arithmetic Data types Homework 3 Binary arithmetic Answers

Topic 4	Floating point arithmetic

Learning Objectives:

- Represent positive and negative numbers with a fractional part in floating point form
- Normalise un-normalised floating point numbers with positive or negative mantissas
- Add and subtract floating point numbers

Content	Resources
Starter Remind students how numbers are held in fixed point binary format, with a set number of bits before and after the binary point. The largest positive number that can be held with 5 places before the point and 3 after, is 01111.111 = 15.875. If you move the point right to increase the **range** of numbers, the **precision** with which a number can be held decreases. **Main** **Floating point numbers** Explain how floating point numbers are held. Students may be familiar with this from multiplying 2 large numbers on a calculator. 75,000 can be represented as 7,500 x 10, or 750 x 10^2, or 0.75 x 10^5, for example. 458.675 could be represented as 0.458675 x 10^3. 0.005 can be represented as 0.5 x 10^{-2}. 0.5 is called the **mantissa**. -2 is the **exponent**. Floating point binary numbers are held in this way, with a **mantissa** and **exponent**. **Converting a floating point binary number to denary** Explain how to convert a floating point number to denary by first moving the binary point the number of places indicated by the exponent. IF the exponent is positive (sign bit 0), the binary point needs to be moved right to make the number larger. **Negative binary to denary**	PowerPoint guide: Data types Topic 4 Floating point arithmetic.pptx

Explain this with the slide. There is an extra slide with a class exercise to translate a number with a negative mantissa into denary. You could give students some extra examples to work out. **Negative exponent** If the exponent is negative, the number becomes smaller rather than larger so the binary point moves left. Give out **Worksheet 4** and ask the students to do the questions in **Task 1**. **Normalisation** This is explained on the next slide, with examples in denary illustrating the point. Give the students time to try normalising the number 0.0000111 0111. Answer on next slide. **Normalising a negative number** More conversions from positive and negative denary numbers to normalised binary are explained. **Negative exponent** With a negative exponent, the binary point must be moved left instead of right. Ask the students to do the questions in **Task 2** of the worksheet. **Adding floating point binary numbers** Show that when denary numbers are expressed in the form 1534 x 10^3, 1025 x 10^2, we cannot simply add the mantissas, if the exponents are unequal. First the exponents must be made equal. With the numbers used in this example, we could do this either by making both exponents equal to 2 or 3, or by expressing the numbers without exponents as 1,534,000 and 102,500. Similarly with binary numbers, we need to equalise the exponents before adding the values. We will do this by first converting to fixed point form (exponents 0). In the example given on the slide, both numbers are positive so a leading zero, the sign bit, needs to be added. **Subtracting floating point numbers** Use the same technique with subtraction. Convert both numbers to fixed point, find the two's complement of the number to be subtracted, and add the numbers. Remember to take the sign bit into	Data types Worksheet 4 Floating point arithmetic Data types Worksheet 4 Answers

consideration. When adding the first number to the two's complement of the second, ignore any overflow bit. Get students to check their results by converting each number to denary and doing the subtraction in denary. In the example given on the slide, A = 13, B = 7. The result of A – B should be 6. Some practice questions are given in **Task 3** of the worksheet. **Plenary** Go over the answers to the worksheet, and the main points of the lesson. Give out **Homework 4**.	Data types Homework 4 Floating point arithmetic Data types Homework 4 Answers

Topic 5	Bitwise manipulation and masks

Learning Objectives:

- Perform logical, arithmetic and circular shifts on binary data
- Perform bitwise operations AND, OR and XOR
- Use masks to manipulate bits

Content	Resources
Starter Explain that this lesson is all about how to manipulate individual bits in a byte. Show the first slide and ask what arithmetic operation would be performed by shifting all the bits one place right, filling in with a zero on the left? This is equivalent to division by 2. To multiply by 2, shift left instead of right. Many processors carry out multiplication and division using a combination of shifts and addition. **Main** **Logical shift instructions** Show how a logical shift right works. The bit that is moved out goes into the carry bit, which can be tested. If it is 1 after a shift right, what does that tell you? It indicates that the number was odd. After two more right shifts, the byte will contain 0000 1001, carry bit 1. Shifting left, it is easier to visualise the carry bit on the left – a good example of abstraction! After shifting a second place left, the result will be carry bit 1, byte 11110100. A logical shift can be used to test the contents of a particular bit, by examining the carry bit after a given number of shifts. Ask, how could you examine the 2nd bit from the left? Answer: Shift left twice, and do a conditional branch depending on the value held in the carry bit. Such instructions are included in the assembly code instruction set of a typical processor. Logical shifts cannot be used for multiplication or division, because the sign bit may change when the bits are shifted. We need another kind of shift – the **arithmetic shift**. **Arithmetic shift instructions** With a right shift, the bit that moves in on the left will be the same as the current most significant bit (msb),	PowerPoint guide: Data types Topic 5 Bitwise manipulation and masks.pptx

thus preserving the sign bit. This will perform integer division by 2, rounding the number down. This can be verified by converting the before and after binary values to denary. 11001111 = -49 11100111 = -25. With a left shift, the sign bit is left unaffected. (You can visualise the carry bit covering it while the shift is made so the 2nd bit moves into the carry bit instead of into the sign bit.) After a second shift left, the carry bit will remain as 1, and the byte will contain 0111 0100 After a third shift, the carry bit will remain as 1, and the byte will contain 0110 1000. **Using shifts to multiply and divide** The next two slides show how multiplying numbers can be done using shifts. (Note that if you start with a large number such as 01111101, as on the "Arithmetic shift left" slide, the shift does not have the simple effect of multiplying the number by 8 because overflow occurs when a 1 is shifted into the carry bit.) Give out **Worksheet 1** and ask students to do **questions 1 and 2 in Task 1**. You can give students time to practise this with more multiplications – make sure the result is not greater than 127 or overflow occurs. **Circular shifts with carry** There are two types of circular shift – one which consider the carry as a "ninth bit" which is included in the circular shift, and another which moves the msb to the lsb (or vice versa) as it shifts all bits in a circular fashion, and simply copies this bit into the carry bit. We will consider only the first type. Ask students to do **question 3** on the worksheet. **Logical instructions** Logical instructions include AND and OR which the students should be familiar with from writing selection statements in their programming exercises. XOR may not be familiar if they have not yet studied Boolean logic and truth tables. It is the exclusive OR, which is true if A OR B is true, but false if both A and B are true. Go through the three operators and explain the concept of using a **mask**. This is a byte containing a pattern of 0s and 1s which in combination with a logical operator, will clear, set or toggle particular bits. To **clear** a particular bit, use a mask of 0 and the AND	Data types Worksheet 5 Bitwise manipulation and masks Data types Worksheet 5 Answers Data types Homework 5 Bitwise manipulation and masks

operator. To **set** a particular bit, use a mask of 1 and the OR operator. To **toggle** a particular bit, use a mask of 1 and the XOR operator. **Plenary** Set **Task 2** on the worksheet. Go over the main points of the lesson, and hand out **Homework 5**.	Data types Homework 5 Answers

Unit assessment

Learning Outcomes:

Students will

- apply their knowledge in answers to a range of questions
- be able to identify areas of strength and any gaps in their understanding of data representation

Content	Resources
Students should complete the **Assessment Test**. The test has been designed to be printed and answered by hand.	Data types Final assessment.docx Data types Final assessment Answers.docx

Acknowledgements

The authors and publisher would like to thank all contributors for their kind permission to reproduce their photographs or images, screenshots of their websites or other copyright material in the PowerPoint Guides.

Every effort has been made to trace the copyright holders and we apologise in advance for any unintentional omissions. We would be pleased to insert the appropriate acknowledgement in any subsequent edition of this publication.

This material contains links to relevant websites featured in the teacher's guide. Every effort has been made to ensure that at the time of distribution, the links remain unbroken, the material remains up-to-date and that links are not inadvertently linked to sites that could be considered offensive. PG Online cannot be held responsible for the content of any website mentioned in this material. It is sometimes possible to find relocated sites by typing the original URL into a browser. Any errors should be reported directly to support@pgonline.co.uk and changes will be made in any subsequent editions of the material.

Artwork

Atlantis © 2011 Hilary Turnbull

Mixed media on canvas, 61x61cm

www.hilaryturnbull.co.uk